Multiple-Object Tracking in the Infrared

Mark Calafut
Stanford University
Electrical Engineering 368
mcalafut@stanford.edu

Abstract — This paper outlines the creation of two multiple-object tracking algorithms for infrared image sequences. The first algorithm uses background estimation to identify foreground objects in individual image frames. Foreground objects are then analyzed to find associations with previously detected foreground objects to maintain a tracking list. Uncertainty is incorporated and used to determine if objects have left the image plane. The second algorithm attempts to improve upon the performance of the first algorithm by using a boosting framework to combine several location and image properties such as convex hull size, multiple correlations, and aspect ratios into a predictor of track associations. The performance of track algorithms one and two was analyzed quantitatively and qualitatively using video sequences from the OTCBVS database. The first algorithm demonstrated effective performance in simplified tracking scenarios but showed limited robustness to occlusion and increasing scene complexity. In preliminary testing, the second algorithm was able to show significantly improved tracking performance during complex scenarios at the expense of increased computation time.

I. INTRODUCTION TO OBJECT TRACKING

Object tracking has long been recognized as an important task in many computer vision and image processing applications. Specifically, object tracking is often involved in automated surveillance of pedestrians or vehicles, in automatic video storage systems, in traffic monitoring, in vehicle navigation, and in motion based recognition algorithms. [1]

The ultimate goal of object tracking is to provide the location of objects in the image plane of each frame of a video sequence. This involves maintaining consistent labeling for each discrete object in the image plane despite movement and potential changes in object appearance. Depending on the tracking algorithm being considered, a variety of information beyond object location can also be generated during the video analysis process. This can include the sizes and shapes of the objects, the changes in the color or the intensities of the objects, and also the motion profiles of the objects.

In general, the object tracking process can be conceptually (and often literally) broken up into discrete subcomponents that are then handled by different image processing techniques. For example, the detection of objects of interest is a highly different process from the tracking of those objects in subsequent video frames. Also, the generation of object representations during tracking analysis requires very different techniques from the logical analysis involved in subsequent comparison of object representations and maintenance of tracking lists. These conceptual divisions make it possible to design modular tracking subcomponents that can be thought of as independent building blocks of the tracking process. In different applications, these separate building blocks have the potential to be interchanged with one another to create a tailored tracking technique for that application. Also, this inherent modularity allows for independent development in subcomponents to be easily integrated into the overall tracking framework. For example, developments in data analysis techniques initially unrelated to object tracking, such as improvements in neural networks can be applied to the object comparison subcomponent of the framework. This new data analysis technique is then easily integrated into the tracking framework allowing for rapid investigation of its affect on overall track performance. This specific paper focuses on developing effective object tracking techniques for infrared video sequences of pedestrians and vehicles, while maintaining sufficient modularity to incorporate alternative subcomponent methods in the future.

II. BACKGROUND: AN OVERVIEW OF A GENERALIZED TRACKING FLOW

To provide context for the algorithms that will be presented in Section III., a generalized object tracking process flow is described in the following paragraphs. To begin the tracking process, first a detection algorithm identifies objects of interest in the first frame of the video and passes their locations to an object analysis algorithm. A variety of techniques exist for the object detection, including image segmentation through mean-shift [2] or graph-cuts [3], foreground separation through background subtraction [4] or through motion template models [5], the use of human input ground truth data, and point detection using Harris corners [6] or SIFT features [7]. These techniques vary in complexity and applicability based on the track application and on the recording conditions. Sometimes the detection process is performed separately from creating a detailed object representation, and in other cases the object representation is used to analyze if potential objects are detections. Regardless of whether the object representation is created during detection or using the detected image patch, a variety of representation types exist for track objects. A few examples of these representation types are point...
representations [8], geometric shape representations [9], skeletal models [10], and single-view templates [11]. Ideally, a generalized tracking algorithm is capable of handling a combination of these representations allowing for a more complex analysis of object associations.

After object representations have been generated for each detection in the first frame, the track list maintenance algorithm assigns separate labels to each track object. The track list can also be used to store important information about the object that will later be used in object association, including the object location (as part of a motion history of the object), the object representation determined by the analysis algorithm, statistical information about the distribution of the object, and filtered or morphologically processed subsections of the patch image.

At this point the tracking program has finalized the initial track list and proceeds to the subsequent frame. If there are any new object detections, handoffs are received from the detection algorithm. The handoff may or may not identify which of these detections are associated with new objects. Following a similar procedure to the one outlined previously, object representations are generated for each of these detections. At this point all image patches of interest in the current frame are identified, and a logic algorithm is used to associate these patches with objects that that are on the track list. The algorithm has available whatever information was stored about the object from previous frames and must compare this information to information extracted from the image patch of interest. This association process is what defines a tracking technique.

Many methods exist for performing the track association process, including statistical methods such as the Kalman filter tracking [12], distribution density based methods such as mean-shift tracking [13], and object representation based matching methods such as shape based tracking [14]. In the case of Kalman filter tracking, predicted object locations are generated through trajectory estimation based on the recorded motion history of the object. The motion of the object can be assumed to be linear, or if more complex motion profiles are expected, then least squares regression can be used to fit a profile to the location history of the object. Ultimately the predicted object location in the image space is compared to the detected object location, and a statistical framework is used to improve the predictive ability of the model for that object. In contrast, other object association techniques, such as the previously mentioned shape based tracking, make little to no use of trajectory information and instead focus on consistency in the shape of objects to perform association. To increase robustness, it is also possible to combine different association frameworks together at the expense of overall computation time. The framework for such an approach is established in Track Algorithm II, presented in Section V.

However, even beyond the choice in performing spatially based association, temporally based association, or combined association, a logical reasoning mechanism also must be designed to implement the association. Frequently these reasoning algorithms make assumptions about changes in the object between frames to facilitate the association process. For example, in the widely used Lucas-Kanade optical flow techniques, it is assumed that the brightness of the object will be constant from frame to frame and that the object will be detectable in a small window around its current or predicted location [15]. These assumptions are used to associate points from one frame to the next, and also to eliminate track points that are no longer visible in the image plane. In many complex scenarios the assumptions made in the Lucas-Kanade technique are not reasonable and will lead to highly inaccurate tracker performance. In these cases, different reasoning techniques must be applied to associate track objects and to eliminate unassociated track objects. Following the association of track points, the track list is updated to include the newly determined position of each object in the current frame. Required spatial or temporal information about that object is also stored in the track list, and unassociated objects can be removed. In subsequent frames this comparison, association, and update process continues, ideally maintaining a track list of all objects present in the image plane. Overall, there is incredible variety in the possible mechanisms that can be employed at each step of the tracking process. The complexity of these different methods also varies greatly, implying a tradeoff between computation time and algorithm robustness. The appropriate level of complexity is ultimately driven by the application of interest and the challenges associated with that application.

III. PROJECT OVERVIEW

This project looked to apply the generalized tracking framework referenced above to the problem of multiple-object tracking in infrared images. Multiple-object tracking in the infrared is of particular importance in surveillance and vehicle navigation applications. The infrared is specifically targeted as a detection band of choice in these applications because heated objects such as pedestrians and vehicles emit consistent infrared signatures during both the night and day. Additionally even in daytime conditions, infrared images are sometimes subject to less background clutter than visible images. However, despite these advantages infrared tracking also exhibits specific challenges beyond that of visible image tracking. For example, visible cameras frequently incorporate multicolor detection at very high resolution. Infrared cameras more frequently detect only a single spectral band at lower resolution. This precludes the use of color based tracking techniques and limits the information available to the track algorithm. Additionally, although the frequency of clutter in infrared images is sometimes reduced in comparison to visible imagery, the type and distribution of the clutter is changed. This makes the rejection of clutter a different process in the infrared than it is in the visible.

After consideration of available data sources, the Object Tracking and Classification in and Beyond the Visible Spectrum (OCTBVS) Benchmark Dataset Collection was identified as an ideal test set for developed infrared track algorithms. Specifically, Dataset 1: OSU Thermal Pedestrian Database and Dataset 5: Terravic Motion IR Database were
selected for testing. Dataset 1 included 10 image sequences of multiple pedestrians walking on the Ohio State University Campus taken from a Raytheon 300D camera mounted on the roof of an eight story building. [16] Ground truth data were available for this dataset. Dataset 5 included 18 image sequences involving a variety of outdoor object motion scenarios taken with a Raytheon L-3 Thermal Eye 2000AS camera. Ground truth data were not available for this dataset. All images in both datasets were in 320x240 pixel format.

Following background estimation, each image sequence was preprocessed. The recorded background template was subtracted from the current frame to identify foreground elements. The foreground image was then thresholded using an experimentally defined threshold. Erosion of the foreground was performed to remove small clutter sources. The size of the rectangular erosion element was experimentally established to achieve optimal detection over the OCTBVS Dataset 1. Following the erosion process, the foreground was dilated with an experimentally determined rectangular structuring element. This structuring element had greater vertical length than horizontal length to better fill the midsections of identified pedestrians. A majority filter was then considered to further fill the interior of target objects. Following this preprocessing, the image was labeled and bounding boxes were extracted for all remaining objects. Detected objects above a threshold size were then compared with track list objects to identify associations. Each object on the track list included a linearly estimated position based on the change in the location of the track object in the previous two frames. The locations of detected images patches from the preprocessing operation were compared to the predicted locations. Detected image patches were then associated with the object at the closest predicted track location. If no track objects existed within an experimentally determined threshold distance from the object, the detected image patch was included as a new track object. All newly defined track objects started with a moderate value for the uncertainty parameter. Each time the track object was successfully associated in a future frame, its uncertainty was decreased by a predefined amount. If the track object was not associated in subsequent frames but had low uncertainty, then the track location for the object was placed at the predicted object location and its uncertainty was increased. This specific mechanism was included to deal with temporary occlusion of objects. If instead the track object had high uncertainty and no association was determined, the track object would not be predictively tracked and would be removed from the track list.

**Figure 1:** An example image from the OCTBVS Database. (Image 31 from Image Sequence 1 of Database 1) [16]

**Figure 2:** Representative tracking in a complex scene from the OCTBVS database [16] using Single-Track. Most track objects are maintained successfully, however two track boxes on the top left have merged.

### IV. Track Algorithm I: Single-Track

Track Algorithm I was designed as a simplified approach to multiple-object tracking incorporating image preprocessing and background subtraction, simple trajectory estimation, and object uncertainty based on the frequency of successful detections. The technique is referred to as Single-Track because it implemented a tracking process on single color IR data. The following paragraphs provide an overview of Single-Track and its tracking performance based on tests using the OCTBVS database.

After analyzing the OCTBVS Dataset 1, it became clear that pedestrian figures were identifiable by their significant positive or negative contrast from the background. To exploit this image characteristic, a background estimation technique was developed. In all frames with detected objects, background was identified using frame to frame subtraction. Following frame subtraction, positive contrast image sections were ignored, and negative contrast sections were recombined with the first of the original two frames. This process provided a reasonable background estimate for any given frame that included detected objects. In frames that did not include detected objects, the entire frame was taken as representative background. Throughout the image sequence, the extracted background from each frame was proportionally averaged with the previously defined background to create a background template of increasing accuracy throughout the image sequence.
Objects with predicted locations near the image boundary were given a lower uncertainty threshold for removal. This was considered representative of their increased likelihood of having left the image plane. After the track association process was completed, tracks that had decreased below a threshold size were removed from the tracking list. The algorithm then displayed the current image with superimposed track boxes shown in red. Track boxes for higher uncertainty targets were displayed in yellow. A mean-shift based supplement was also added to the algorithm. This algorithm was used to identify the distributional centers of identified track patches and was displayed as a blue box in the image plane. This algorithm can also be used to identify objects in place of the currently used image labeling method, but this modification did not lead to performance improvements.

The creation of the Single-Track algorithm was an iterative process. Although it was designed as a simplified tracking method based on effective image preprocessing, it became necessary to use a complex logical flow based on uncertainty to determine if unassociated track points should be removed or predictively tracked. The algorithm also involved a series of image processing parameters including erosion and dilation, element sizes, and threshold values. A Matlab script titled SetParm.m was developed to optimize the values of these parameters over the first video in Dataset 1. The optimization was performed by iterating through parameters values and finding the combination of values that minimized the total error between each ground truth object location and the closest recorded track point. Optimization was also performed to determine the parameter values that minimized the error in number of track points in the scene compared to the number of track points in the ground truth data. The minimization of error in the number of track points was considered to be a better predictor of overall track performance and was selected for future testing. Testing Single-Track on other videos in the test set showed that although its performance was relatively effective in simple scenes, its tracking strategy was not highly robust. The specific structuring elements selected during the optimization process sometimes combined objects of interest into single larger detections, rather than connecting interior components. To determine if Single-Track could be better optimized over the full Dataset 1, SetParm.m was reconfigured to optimize the parameters over the entire dataset. The optimized values are shown in Figure 5. Overall, the performance of the algorithm was effective in less complicated scenes, particularly when objects were not overlapping. However despite the attempts to optimize the algorithm, it would make errors while tracking objects in complex scenes. As a final note, large amounts of raw data were collected in the Single-Track optimization process that are not easily presented in this report. Appendix A contains a sample of this data to demonstrate its formatting and style.

For Dataset 5, no quantitative ground truth data were available for object location. After some parameter adjustment, testing of Single-Track with the IR sequences showed effective qualitative tracking performance. The parameter values used in tracking were also the values shown in Figure 5. Objects were usually successfully identified and track boxes maintained close proximity to the targets. The improved qualitative performance of tracking Dataset 5 as compared to Dataset 1 can be attributed to the reduced complexity of many of the scenes in Dataset 5. Dataset 5 sequences usually involved smaller numbers of pedestrians and greater object separation. However in cases where Dataset 5 sequences involved significant occlusion of targets for long periods of time, Single-Track would also demonstrate merging.
of track boxes as occurred in sequences from Dataset 1. In general, Single-Track is a suitable technique for applications involving simplified motion scenarios or for applications that require reduced computational complexity.

V. TRACK ALGORITHM II: NODETECT-TRACK

Track Algorithm II was developed to specifically address the weaknesses of the Single-Track algorithm by incorporating greater spatial information in the association process. The Single-Track Algorithm relied very heavily on preprocessing to eliminate clutter and on temporal information to estimate object position. Although it was able to successfully track individual or well separated objects, it tended to struggle on tracking objects that were close to one another. Track Algorithm II, referred to as NoDetect-Track was designed to analyze additional information related to the appearance of track objects. This allowed for comparison of not only patch locations but also of image properties between objects of interest.

NoDetect Track was designed to completely separate the object identification task from the tracking or object association task. Detections were provided by prepared ground truth data. Ground truth data from the OCTBVS Database 1 was modified manually to include labels for all newly apparent objects. This ground truth file was then used to pass the detection location of new objects in the first frame that they were visible in the image sequence. Following this location handoff, the NoDetect-Track algorithm was responsible for determining the location of the object in all subsequent image frames. The NoDetect-Track algorithm accomplished this by extracting a variety of image properties for each object in its track list. Specifically it extracted the image patch itself and then calculated the convex area, the extent, the aspect ratio, the orientation, and the perimeter of the thresholded and background subtracted object. If multiple objects were visible in the patch, the features of the largest object were taken to be representative of the true track object. In each subsequent image frame, the algorithm first thresholded the isolated foreground image and then identified all objects in the scene. The morphological processing performed in Single-Track was not performed in NoDetect-Track. For each object in the track list, the correlation of the recorded image patch and the foreground isolated image patch were computed with the new image frame and the foreground isolated image frame respectively. For each detected object in the scene, a comparison was performed between the parameters of the track object and the parameters of the detected object. This comparison was performed using a modified boosting framework that used a variety of measured parameters as weak classifiers. [17] The comparison result was a linear combination of the normalized difference between the object values and detected parameter values for location, convex area, extent, aspect ratio, orientation, perimeter, foreground isolated correlation, and raw image correlation.

\[
V = A_1 \frac{\text{Error}_x}{\text{Size}_x} + A_2 \frac{\text{Error}_y}{\text{Size}_y} + A_3 \frac{\text{Error}_{\text{ConvArea}}}{\text{Value}_{\text{ConvArea}}} + \ldots A_n \frac{\text{Error}_{\text{RawCorr}}}{1}
\]

The weight of each of these parameters was set through experimental testing. The estimated optimal weights can be found in Figure 8 on the following page. After calculating these normalized differences, the detected object with the minimum weighted error was associated with the appropriate object on the track list. If the minimum weighted error value for this object exceeded an experimentally defined threshold (V_Thresh), it was assumed that the track object had left the field of view. Location prediction and background updating were performed using the same mechanisms as were implemented in Single-Track.

The determination of the boosting weights for NoDetect-Track was performed using Image Sequence 1 in OCTBVS Dataset 1. Based on our work performed with Single-Track, the error in number of detections for each video frame was chosen as the evaluation parameter for optimization of each weight parameter. Due to time limitations, only preliminary quantitative testing could be done using NoDetect-Track. Entering ground truth labels for the image sequences was
Table: Preliminary optimal parameters based for the NoDetect-Track algorithm over Image Sequence 1 of Database 1. A1 to A11 refer to linear weights in calculation of an association value. V_Thresh is the maximum value at which points are still considered associated. Optimization was performed using SetParmNDTA.m.

![Table](image)

extremely time consuming, as was optimization of the thirteen parameters over a reasonable range of values.

However even considering this limitation, the initial results for NoDetect-Track far exceeded the performance of Single-Track. After initial optimization, NoDetect-Track experienced only one incorrect identification (during only one frame) throughout the entirety of Dataset 1, Image Sequence 1. This high accuracy was achieved despite the presence of seven closely spaced objects in several of the frames. It is also likely that the addition of new weak classifiers and increased optimization of the parameter set would allow for successful identification of all objects in the dataset.

NoDetect-Track also did not suffer from some of the commonly seen difficulties seen with Single-Track. For example, the qualitative merging of track boxes seen in Single-Track did not occur in NoDetect-Track. This was likely due to the incorporation of object appearance into the association process. Figure 6 and Figure 7 on the previous page show successful tracking of spatially close objects from Image Sequence 4 and Image Sequence 1 of Dataset 1. Overall, the initial results using NoDetect-Track are considered highly promising and are only expected to improve as the algorithm is further optimized and expanded. Due to the high difficulty of tracking in the OCTBVS environment, it was expected that NoDetect-Track could also be optimized to work effectively in other challenging infrared image tracking applications. The algorithm also has the flexibility to adjust its parameter values or to include new weak classifiers as applications require.

VI. CONCLUSIONS

The work described in the paper demonstrates the importance of tailoring an image tracking technique to the application of interest. Less computationally intensive algorithms such as Single-Track and mean-shift based tracking can be highly effective at tracking single targets or tracking multiple separated targets. However additional discriminating factors are required to track in more complex scenarios. Qualitative and initial quantitative results using NoDetect-Track demonstrated a significant improvement in tracking performance in these complicated scenarios. Future work would ideally identify additional image patch properties to use as weak classifiers. A systematic consideration of a variety of weak classifiers, and subsequent optimization of the weights of these classifiers, would likely lead to further improvements in performance. Another potential improvement of the NoDetect-Track would involve changing the value of classifier weights during the video sequence. For example, the background estimation method employed becomes more accurate as more frames are processed. It would therefore be reasonable to increase the weight of classifiers that are based on background subtracted data later in the image sequence. A simple mechanism of this type was included in the final version of NoDetect-Track and would be expanded in future versions of the algorithm.

VII. REFERENCES